National Research University Higher School of Economics Faculty of Computer Science

Programme' Master of Data Science'

MASTER'S THESIS

Detecting patterns in purchase history using association rule learning methods

Student:

David Schmid

Supervisor:

Anastasia Maksimovskaya

Moscow, 2023

ABSTRACT

Based on association rule mining, the thesis suggests a solution to find more relevant and stronger itemsets and associative rules by "Reducing the number of passes over the database," "Sampling the database," "Adding extra constraints on the structure of patterns" and "Increase efficiency through parallelization." It uses FP-growth as a baseline algorithm and introduces profit as a new metric for filtering and comparing the association rules. The experiments show that the modified FP-growth algorithm can generate more relevant and stronger rules.

Table of Contents

[ABSTRACT 2](#_Toc123116741)

[1. Introduction 5](#_Toc123116742)

[2 Discussion of related works 8](#_Toc123116743)

[3 Baseline algorithm 12](#_Toc123116744)

[3.1 Simple dataset 12](#_Toc123116745)

[3.2 Formal model of association rules 13](#_Toc123116746)

[3.2.1 Support 13](#_Toc123116747)

[3.2.2 Paths: 14](#_Toc123116748)

[3.2.3 Confidence: 14](#_Toc123116749)

[3.2.4 Application of association rules to the simple dataset 15](#_Toc123116750)

[3.3 A-priori 17](#_Toc123116751)

[3.4 FP-growth 19](#_Toc123116752)

[4 Developing a modified Algorithm based on FP-growth algorithm 24](#_Toc123116753)

[4.1 FP-growth Model adjustments 24](#_Toc123116754)

[4.1.1 Introduction 24](#_Toc123116755)

[4.1.1 Date-decay-function 25](#_Toc123116756)

[4.1.2 Profit function 26](#_Toc123116757)

[4.2 Modified FP-growth reconciliation with TAR 28](#_Toc123116758)

[4.3 Modified FP-growth with date decay 30](#_Toc123116759)

[4.4 Modified FP-growth with profit 33](#_Toc123116760)

[4.5 Modified FP-growth with combined date decay and profit 38](#_Toc123116761)

[4.6 Performance, Strengths, and weaknesses of the author's modified FP-growth algorithm 40](#_Toc123116762)

[5 Experiments/ Analysis of big datasets 42](#_Toc123116763)

[5.1 E-Commerce purchase history from an electronic store 42](#_Toc123116764)

[5.1.1 Dataset 42](#_Toc123116765)

[5.1.2 Analysis 43](#_Toc123116766)

[5.2 E-Commerce purchase history from a jewelry store 46](#_Toc123116767)

[5.2.1 Dataset 46](#_Toc123116768)

[5.2.2 Analysis 48](#_Toc123116769)

[5.3 E-Commerce purchase history from a cosmetics store 51](#_Toc123116770)

[5.3.1 Dataset 51](#_Toc123116771)

[5.2.2 Analysis 53](#_Toc123116772)

[6 Conclusion 55](#_Toc123116773)

[7 References 57](#_Toc123116774)

[8 Applications 59](#_Toc123116775)

1. Introduction

The concept of association rules (AR) was popularized due to a 1993 article by Agrawal et al. [1]. This article shows the application of AR, namely the "A-priori-algorithm" for shelf management to maximize profit. In the evolution of basket data mining, the application of AR belongs to the early methods, which are simple but powerful. AR has not lost its significance in the research. More efficient and modified versions of the base algorithm are developed to mine the relevant and significant basket relations more efficiently. Compared to more complex ML or DL models, the most significant advantage of this relatively simple algorithm is its ease of use and generalization of pre-defined rules, making it scalable to millions of items and Big Data. Moreover, it is simple enough to explain to businesses conceptionally.

The main driver for the business is ultimately not the frequency of items but the profitability. One can assume that more frequent items are lower in price and higher in the margin but can be equal with not frequent but highly-priced articles. In an Online-Article from Chou, T [15], a conclusion to AR is written the following:

"I would like to share an interesting story. While I was writing this article after work, a full-time engineer walked by, and he saw that I was writing something about A-priori and Fp Growth. He said, "Interesting, but not realistic." He further explained that this algorithm doesn't take weighing under consideration. For example, what about a transaction with multiple same items? There are also more subtle conditions that are not included in this algorithm. And that's why companies won't implement this in their business."

That conclusion is not in academic language, but it concludes the real-world experience with AR. The author's thesis wants to address that issue, that it is not frequently implemented in real business, and especially proposes a solution for a consideration of transactions transaction with multiple same items.

Frequency is only one part, but not the ultimate driver for business. The interference of profit and frequency could be solved by post-application of associative rules when connecting all frequencies with profit from articles later. However, in traditional association rules (TAR), we have a **first problem**: we already sorted out the least frequent items and lost crucial relations.

Moreover, so this approach would not be that straightforward. Instead, the author suggests considering profit already within the algorithm as decision criteria, whether to keep the item for getting relational rules.

**The second problem** not considered in TAR is that items can occur multiple times within one transaction. The support counting does not work in such a scenario. Even such a scenario is dependent on the nature of items quite frequently.

The author suggests adding the profit of these items and dividing by the unique count of these items, leading to a new average profit per item. In such a way, we consider these weights in the profit but not in the count, keeping the AR's primal structural integrity based on simple counts of each item per transaction.

**A third problem** poses the question: Why should we consider all transactions with the exact counting if more recent items have higher relevance than older items?

The author suggests introducing a date-decay function to consider the relevance of more recent items.

**The fourth and last problem** the author addresses is algorithm efficiency. We are primarily interested in the itemsets themselves; instead of repetitions leading to the same items. Even the improved FP-Tree algorithm has to loop through the whole dataset many times to find the exact relation between the identified itemset branches.

The author suggests that by simplifying counting directly from the original tree structure (one loop through the whole dataset), we can derive the complete itemsets already with the most count to the minor count of items (1 path). Mostly the other paths are repetitions and do not have many gains. That approach will make the algorithm super fast, with some loss of information (loss of other less relevant paths).

2 Discussion of related works

AR was popularized and, in effect, introduced to the public due to a 1993 article by Agrawal et al. [1]. The articles describe the advances in barcode technology and the recording of transactions. Before that, a computer-recorded transaction was not even possible. According to Agrawal et al. [1], the analysis aims to "how to place merchandise on shelves to maximize the profit." They introduced a formal association rule model.

Interestingly this model has not named an A-priori algorithm yet but has been labeled as that after their paper. Fundamental terms like antecedent, consequent, resulting itemset, and support are introduced. Agrawal et al. [1] already optimized how to find association rules. Instead of finding all possible candidates for an association rule, it can be defined if at least one of the children (paths) of a new itemset candidate is not frequent, the resulting itemset will for sure not be frequent anymore, limiting the number of possible itemset candidates and improving the performance. Agrawal et al. [1] make the following limitation: "The algorithm proposed in this paper is targeted at discovering qualitative rules. However, the rules we discover are not classification rules. We have no pre-specified classes. Rather, we find all the rules that describe the association between sets of items." In the author's opinion, that is a crucial statement because to find relevant items, the dataset should be pre-processed in categories to find the relevant connection in each classification; otherwise, the algorithm would not be able to handle the sheer mass of associations, if the support-parameter is chosen too low. Details about the fundamental A-priori algorithm will be presented in chapter 3.

Some years later, in a paper called "Association Rules Mining: A Recent Overview, "Kotsiantis, S. and Kanellopoulos, D (2006) [5] state: "In many cases, the algorithms generate an extremely large number of association rules, often in thousands or even millions. Further, the association rules are sometimes very large. It is nearly impossible for the end users to comprehend or validate such large number of complex association rules, thereby limiting the usefulness of the data mining results." They state that there are four main directions of research to address this issue:

1. by reducing the number of passes over the database
2. by sampling the database
3. by adding extra constraints on the structure of patterns
4. by through parallelization.

The first point is addressed by the much more efficient FP-growth algorithm developed by Han, J. and Pei, J. (2000) [6], breaking the bottleneck of the A-priori algorithm. Toivonen, H, has developed on the second point. A sample is used to represent the whole dataset. Therefore, the choice of support level has to be much lower to find the same association rules as in the whole dataset. However, the main problem here is what sampling method should be chosen. Sampling Error Estimation (SEE) plays a key role and has to match a certain confidence level. The third point can be implemented as pre-processing the source file, filtering during the algorithm, or post-processing. The key in all methods is to find the "relevant patterns." Rapid Association Rule Mining (RARM) is an association rule mining method developed by Das, A. (2001) [8] is an association rule mining method that uses the tree structure to represent the original database and avoids the candidate generation process. According to Das, A., the bottleneck is the 2-candidate itemsets because the number of candidates in A-priori are all combinations of items that are greater than the minimum support, meaning

Das, A. (2001) [8] states that the algorithm is up to 100 times faster than the A-priori algorithm. It can be seen as a predecessor of the FP-growth algorithm, entirely relying on a tree structure. The fourth point is the most technical and is used to improve an already existing algorithm; there is much research on the parallelization of A-priori and FP-growth algorithms. That is especially important when using the algorithm in production and strict time and efficiency constraints.

In a paper by Slimani, T. El al. (2014) [9], different association rule mining approaches and variants are discussed. It shows that research in association rules was and is quite active.

A choice of the developed methods listed are:  
- A-priori (1994)  
- A-prioriTID (1994)  
- DHP (1995)  
- FDM (1996)  
- GSP (1996)  
- DIC (1997)  
- PincerSearch (1998)  
- CARMA (1999)  
- CHARM (1999)  
- Depth-project (2000)  
- Eclat (2000)  
- SPAD (2001)  
- SPAM (2002)  
- Diffset (2003)  
- FP-growth (2004)  
- DSM-FI (2004)  
- PRICES (2004)  
- PrefixSpan (2004)  
- Sporadic Rules (2005)  
- IGB (2005)  
- GenMax (2005)  
- FPMax (2005)  
- FHARM (2006)  
- H-mine (2007)  
- FHSAR (2008)  
- Reverse A-priori (2008)  
- DTFIM (2008)  
- GIT-tree (2009)  
- Scaling A-priori (2010)  
- CMRules (2010)  
- Minimum effort (2011  
- TopSeqRules (2011)  
- FPG ARM (2012)  
- TNR (2012)  
- ClaSP (2013)

We see that there are a vast number of different approaches. However, all can be categories in one or several of the four main directions to address the performance issue and find the relevant itemsets and rules.

Recent papers try to generate association rules based on DL. 10. Patel, H, K and Yadav, K, P. (2022) [10] show the increased efficiency of an autoencoder algorithm called DAENMF-ARM, a" denoising autoencoder and non-negative matrix factorization based on association rule mining" to increase the efficiency compared to the classical baseline algorithms A-priori, Eclat, and FP-growth.

Al Shehabi S. et al. (2021) [11] introduce an algorithm called MARC to extract association rules, which is based on a Multi Self-Organizing Map (MultiSOM). The goal is to increase efficiency and only find relevant association rules.

The author spent some time rebuilding some new metrics suggested by Bao, F. et al. [12]. They suggest introducing new association rules and measures for validation, called Bi-support, Bi-lift, Bi-improvement, and Bi-confidence. They consider the correlation of non-support. If item A has a support of 1000 and item B has a support of 100, and they have joint support of 100 as itemset, then item A is just a very popular overall item but has no direct correlation to B. That can be found with these new measures.

3 Baseline algorithm

## 3.1 Simple dataset

For demonstrating the method of the basic associative algorithm, a simple dataset, the author uses the following dataset. The dataset is has chosen to be simply enough to explain the concept of the algorithms. The dataset consists of 10 items, denoted as and 10 Transactions as .

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Index** | **I0** | **I1** | **I2** | **I3** | **I4** | **I5** | **I6** | **I7** | **I8** | **I9** | **I** |
| T0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 7 |
| T1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 7 |
| T2 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 4 |
| T3 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 7 |
| T4 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 5 |
| T5 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 5 |
| T6 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 3 |
| T7 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 4 |
| T8 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 3 |
| T9 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 3 |
| **T** | **8** | **7** | **4** | **4** | **5** | **3** | **2** | **4** | **6** | **5** | **48** |

Figure 1: Simple dataset

## 3.2 Formal model of association rules

The author takes the terms and general description of the association rules model by Agrawal et al. [1].

Let I be the   
Let   
Each item is binary. If an item is in the itemset, it is denoted by 1; if missing by 0.  
Let T be the database/dataset of transactions.  
Let   
Let X be a set of items in each transaction.  
A transaction Tn satisfies X if all X items commonly occur in Tn.

3.2.1 Support

The number of elements of X is 10.  
1. If .  
T satisfies X in 8 transactions.  
Therefore the support of X is 0.8.  
2. If .  
T satisfies X in 7 transactions.  
Therefore the support of X is 0.7.  
3. If .  
T satisfies X in 5 transactions.

If X consists of at least two items, we have two paths to get to itemset X.  
Agrawal et al. [1] name the existing itemset "antecedent" and the newly added item "consequent."

3.2.2 Paths:

Suppose we have an itemset with support 2. The number of paths leading to an itemset is factorial n!. Therefore for 2 items, there are 2 paths.

First path:, where I0 is the antecedent, and I1 is the consequent, building the itemset X.  
Second path:, where I1 is the antecedent, and I0 is the consequent, building the itemset X.

### 3.2.3 Confidence:

The metric confidence is calculated by dividing the itemset's support by the antecedent's support.

First path's confidence:  
Support of ; Support of

Second path's confidence:  
Support of ; Support of

We see there is not the same confidence for both paths, even if they lead to the same itemset X.

### 3.2.4 Application of association rules to the simple dataset

We can construct and filter out all association rules depending on the itemset's minimum support and confidence.

Suppose we want to find all itemsets with:

|  |  |  |
| --- | --- | --- |
| **Index** | **Itemset** | **Support** |
| 0 | (I0) | 0.8 |
| 1 | (I1) | 0.7 |
| 2 | (I8) | 0.6 |
| 3 | (I4) | 0.5 |
| 4 | (I9) | 0.5 |
| 5 | (I0, I1) | 0.5 |
| 6 | (I0, I8) | 0.5 |
| 7 | (I1, I9) | 0.5 |

Figure 2: Simple dataset - support

For illustration, the author chose enough high support min\_support of 0.5 to filter only six frequent itemsets. If there is no min\_support, there will be 308 itemsets. No restrictions on confidence are applied.

Generated association rules:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Index** | **Itemset** | **Support** | **Ant.** | **Ant. Supp.** | **Cons** | **Cons. Supp.** | **Confidence** |
| 0 | (I0, I1) | 0.5 | (I1) | 0.7 | (I0) | 0.8 | 0.714 |
| 1 | (I0, I1) | 0.5 | (I0) | 0.8 | (I1) | 0.7 | 0.625 |
| 2 | (I0, I8) | 0.5 | (I8) | 0.6 | (I0) | 0.8 | 0.833 |
| 3 | (I0, I8) | 0.5 | (I0) | 0.8 | (I8) | 0.6 | 0.625 |
| 4 | (I1, I9) | 0.5 | (I9) | 0.7 | (I1) | 0.5 | 0.714 |
| 5 | (I1, I9) | 0.5 | (1) | 0.5 | (I9) | 0.7 | 1.000 |

Figure 3: Simple dataset – association rules

When choosing confidence as min threshold, only itemsets of at least two items get filtered out because single items have no confidence. The confidence threshold has been chosen to be small enough to show all possible combinations of the three frequent itemsets with two items:

{I\_0,I\_1},  
{I\_0,I\_8},  
{I\_1,I\_9}

Two paths exist for each of the three itemsets of precisely two items; in total, 3 \* 2 = 6 possible combinations.

If no minimum support is chosen, 308 itemsets build the basis for the rules. If no confidence threshold is chosen, these 308 itemsets generate 5,183 rules. Considering the straightforward dataset T, introduced in chapter 2, is overwhelming. That reflects problem 1 (relevance) and problem 4 (efficiency and order) described in the introduction.

## 3.3 A-priori

A-priori, Eclat, and FP-growth, all the TAR algorithms, result in the same frequent itemsets and rules. There is no difference in the outcome, but there is a difference in the efficiency. A short example shall illustrate the difference between the least efficient A-priori Algorithm and the most efficient FP-growth algorithm.

The author recalculates the outcome from the Python library (A-priori) step by step.

**Step 1:**  
Let us consider all frequent itemsets >= 0.5 minimum support.  
{I\_0} Support: 0.8  
{I\_1} Support: 0.7  
{I\_4} Support: 0.5  
{I\_8} Support: 0.6  
{I\_9} Support: 0.5  
That we call **List1**  
Notice: For the first round, one loop in the database is enough; count all occurrences of all items and compare with minimal support

**Step 2:**  
Building all possible combinations out of List1 gives back the following candidate itemsets:  
{I\_0,I\_1},{ I\_0,I\_4},{I\_0,I\_8},{I\_0,I\_9}  
{I\_1,I\_4},{I\_1,I\_8},{I\_1,I\_9}

{I\_4,I\_8},{I\_4,I\_9}  
{I\_8,I\_9}

**Step 3:**  
Repeat Step 1 for candidate itemsets 1 >=0.5 minimum support  
{I\_0,I\_1} Support: 0.5  
{I\_0,I\_8} Support: 0.5  
{I\_1,I\_9} Support: 0.5  
That we call **List2**  
Notice: Instead of simply looping once, we need to check all possible candidates to be a subset of the transactions for every transaction. If their list of candidates grows to several 1,000 and the database has over 1 million rows, it can become very inefficient and slow on an ordinary Desktop computer (without distributed calculations)

**Step 4:**  
From List 2, we can create a new candidate set by joining all possible combinations with a length of itemset three together.  
{I\_0,I\_1,I\_8},{I\_0,I\_1,I\_9},{I\_0,I\_8,I\_9},{I\_1,I\_8,I\_9}

**Step 5:**  
By pruning, which means we build all possible subsets with two elements (1 element less than the candidate itemset), we can figure out if the new candidates already disqualify as not above minimum support. If one of the subsets is not in the already found frequent item list, the resulting itemset will not be frequent either. This way, not all candidates have to be looped in the database.

Let Frequent = F, Not Frequent = NF  
Subsets for {I\_0,I\_1,I\_8}: {I\_0,I\_1}:F, {I\_0,I\_8}:F, {I\_1,I\_8}: NF -> Disqualified  
Subsets for {I\_0,I\_1,I\_9}: {I\_0,I\_1}:F, {I\_0,I\_9}:NF ,{I\_1,I\_9}: F -> Disqualified  
Subsets for {I\_0,I\_8,I\_9}: {I\_0,I\_8}:F, {I\_0,I\_9}:NF ,{I\_8,I\_9}: NF -> Disqualified  
Subsets for {I\_1,I\_8,I\_9}: {I\_1,I\_8}:NF, {I\_1,I\_9}:F ,{I\_8,I\_9}: NF -> Disqualified

All four candidate itemsets contain at least one subset, which is not frequent. Therefore the A-priori algorithm will stop here and give back the frequent itemsets and rules. If there are no constraints to any other metrics, the possible rules per itemset are factorial n!.

If at least one candidate is left, steps 3-5 are repeated until no frequent itemset can be found.

**Time-complexity:** Tahyudin, I. et al. [13] show that the time-complexity of the A-priori algorithm is , while "n" is the total number of all unique items above the minimum support threshold. We see that we will have many problems if the minimum support is too low and all items are interconnected. Indeed in a real dataset, A-priori still can perform okay, but in the worst case, when all items are associated and a too-low minimum support has been chosen, it becomes nearly impossible to proceed with this algorithm.

## 3.4 FP-growth

The FP-growth algorithm is a big step toward a much more efficient solution of finding frequent itemsets. FP stands for "Frequent Pattern," structured in tree nodes, which grow with each new combination and are dynamically incremented.

The most significant change is that each candidate set has to be looped through the whole dataset. Important to notice is that both algorithms, A-priori and FP-growth, do not change the methodology of frequent itemsets and association rules but have different approaches to getting the result more or less efficiently.

**Step 1** is the same as for A-priori:  
Let us consider all frequent itemsets >= 0.5 minimum support.  
{I\_0} Support: 0.8  
{I\_1} Support: 0.7  
{I\_8} Support: 0.6  
{I\_4} Support: 0.5  
{I\_9} Support: 0.5  
This we call **List1**

Notice: In FP-growth, the outcome of the list is kept in strict order by count. If two items have the exact count, it does not matter, but most important, once fixed, the order is kept over the whole algorithm.

In Theory, it would be possible to have a random order, which is fixed, or even the most minor frequent item at the top, and it still would work and lead to the same result. However, we have an upside-down or bizarre-looking tree that would take away some of the efficiency and waste memory and computing power. Therefore there is no reason not to order it from most frequent to least frequent.

**Step 2:**The root of the tree is denoted as null.

Building the tree

T0 is filtered by minimum support and ordered by the strictly ordered list of Step 1=

T0 is filtered by minimum support and ordered by the strictly ordered list of Step 1=

The share the same node I0. Therefore, this node is incremented by 1. I4 is a direct child of I0. This combination still does not yet exist and therefore builds a new branch.

After adding all transactions into the tree, the tree looks like the following:

**Step 3:**  
A recursive function results in the following sets best gather the frequent itemsets:

**Step 4:**  
From the itemsets in Step 3, we can build all the frequent itemsets.  
For example, finding the association rule of I1 -> I9:  
To find the support total of I1, we sum up all generated sets of Step 3, which have I1 as the last item. These are

Afterward, we sum up all generated sets of Step 3, which have I1 in it and I9 as the last item.

Therefore for AR I1 -> I9, the antecedent has the support of 7, and the common itemset has the support of 3.

For finding the opposite association rule of I9 -> I1, in short:  
3

Therefore for AR I9 -> I1, the antecedent has the support of 3, and the common itemset has the support of 3. There is no other I9 besides all the sets in which A1 is commonly present. Therefore, both the antecedent and the itemset have the support of 3.

**Time-complexity:** The maximum time complexity of FP-growth is . [14] That is substantially better than . In small datasets, A-priori and FP-growth will perform both well. However, as soon as there is an extensive dataset, FP-growth will outperform A-priori by far, and A-priori becomes "unusable."

4 Developing a modified Algorithm based on FP-growth algorithm

## 4.1 FP-growth Model adjustments

4.1.1 Introduction

As discussed in chapter 2, there are four main directions of AR development [5]

1. by reducing the number of passes over the database
2. by sampling the database
3. by adding extra constraints on the structure of patterns
4. by through parallelization.

The author's modified FP-growth algorithm's adjustments belong to the second point. We mainly consider the transaction's date and the profit of the transaction during the algorithm. Question: Why does the author not consider it a pre- or post-processing step?

For the date-decay, it is thinkable to do it in a pre-processing step. The date-decay function will change the support of the transaction weight, the more current, the more weight. The function's curve can be adjusted flexibly by a lambda function argument. However, it is much handier to do it inside the algorithm for practical reasons; we only have to think about the parameters, like the date function. Post-processing with a date decay function is impossible because the date information will get lost in the itemsets and association rules. Moreover, date-decay support serves as a filtering criterion for the minimum support.

For the profit function, pre-processing of the data is thinkable, but then we do not have the author's so-called "associated profit" of itemsets, which can be a handy feature to see the overall profit of an item set. It is the most practical way to process the data inside the algorithm. Post-processing is not thinkable because profit-based support serves as a filtering criterion for the minimum support.

Let us discuss the adjustment steps in summary. The details of "the simple dataset" are shown in chapters 4.3 and 4.4. There is no subchapter for the single path modification, but it is explained in chapter 4.2.

4.1.1 Date-decay-function  
Step 1 – Parameters:  
1. date\_col:  
The date column position selection, beginning with 0. 2, will be the third column of the pandas' data frame.  
2. max\_date:  
A dataset with a date range of 2022-01-01 until 2022-01-31 will be 2022-01-31.  
3. date\_range:  
If a dataset has a date range of 2022-01-01 until 2022-01-31, it will be 365.  
4. date\_sensitivity  
A lambda function must be given as an input values parameter [0,1]. For example

The date-decay function is processed inside the algorithm as follows:

Step 2 – Date-decay function:  
for i in range(date\_range):  
 date\_sensitivity(( date\_range – i ) / date\_range))

Therefore the input values will always result in a range [0,1]. The lambda function can be flexibly chosen for each dataset anew, if necessary.

Step 3 – Multiplying the resulting weights with transactions support:

The resulting weights are multiplied with the support of the transaction, directly manipulating the support. That will modify the standard AR methodology and interpretation. Therefore it should be treated with caution.

**Time-complexity:** The timecomplexity is the same as for FP-growth, which is [14]. It should run faster by showing only the most frequent single path. However, it does not change the time complexity fundamentally.

### **4.1.2 Profit function**

In the author's opinion, the profit function is the more powerful and practical function because it solves the fundamental problem of having more than one of the same items in the same transaction.

Step 1 – Parameters:  
1. profit\_col:  
Selection of the profit column position, beginning with 0. 3, will be then the fourth column of the pandas' dataframe.  
2. max\_profit:  
That is the max profit of all items. That is used to convert each item's profit in percentage values.  
3. profit\_sensitivity  
A lambda function must be given as an input values parameter [0,1]. For example

This function will be embedded to:

The lambda function is just a straight linear function and does not weigh the item's profit in any way. However, it could be thinkable to have a non-linear curve for special effects or finding rules, which otherwise could not be found.

Step 2 – Minimum profit for filtering out qualified items for association rules

The overall profit per item is used instead of support (frequency) to decide whether an item qualified as a candidate. That is a primary trigger to find more relevant associated itemsets. Therefore an expensive item with fewer transactions will be qualified for association rules, which dropped out with TAR.

Minimum profit is calculated from the parameter support. Therefore:

The overall profit will be diluted if additional profit and date-decay function are active:

Step 3 – Calculating average profit:  
The profit function is like a separate metric, summing up the profit by the lambda function

of each item in the first part of the algorithm. The difference to support summing up is that we sum up the transaction in the support. In the profit metric, we sum up each item, even the multiple items. However, we need a single count of each item once in a transaction to calculate its average later. For these reasons, we save both profit per item and single count per transaction per item in a dictionary.

In the end, we calculate the average profit like:

Step 4 - Associated profit:

For the FP-growth, we still use the support for TAR. The profit is only a second metric. We calculate the associated profit at the end as follows:

There is tension between supportItemset and avgProfit. The more items we have an in a set, the higher the sum of avgProfit will be. However, most probably, the support for that new itemset will shrink. However, it cannot be predicted if the resulting profit will rise or shrink; the change in profit of a new item added is an interesting metric to observe.

## 4.2 Modified FP-growth reconciliation with TAR

First, the author has to prove that the modified FP-growth, the author's implementation based on FP-growth, will give the same results as the traditional ones.

A screenshot of a computer

Description automatically generated with medium confidence

Generally, the author does not modularize frequent itemsets and rules but gives back both attributes at once. One difference, which can be observed, is that we have only three rules for frequent itemsets with two items instead of 6 rules. Per frequent itemsets, the descending order of total count is taken to show the "most relevant path."  
  
That restriction is a conscious decision. One can argue that it takes away information. However, if we focus on solving business problems and later on focus on profit, this will have the following advantages:

- Generally more efficient algorithm  
- Clearer rules. If there are frequent datasets with many items, the factorial n! will be shown because all subsets are frequent, too. That will spam the conscious analysis and take away the focus. If we do not have minimum support, 308 frequent itemsets will generate 5,183 rules for this specific dataset. Instead, in the modified algorithm, we show only the most relevant path, 308 rules for 308 item sets.  
- With additional elements such as profit metrics, it will become chaotic otherwise and take away the essence and focus.

## 4.3 Modified FP-growth with date decay

The author has a second suggestion, how to improve the traditional association rules. This suggestion refers to the date decay function. We can assume that more recent transactions may have a higher relevance than older ones. That is especially true for products with a very short lifecycle, like consumer electronics. We have two options if the transaction period is two years and the typical product lifecycle is only one year. Either cut the transactions to adjust to the product lifecycle or a smarter one, to have a date decay function. The oldest items weigh 0.7, and the most recent items weigh 1. That is purely speculation, and we must research and find the ideal date decay function for every new dataset. There cannot be only one perfect date decay function for all datasets; instead, it must be calibrated for every dataset again. This date decay function has to be implemented in the frequent items and association rules itself because later, it cannot be calculated anymore.

Let us see how it works with the simple dataset.

Text

Description automatically generated

Graphical user interface, text

Description automatically generated

T1 has the date of 2022-11-01.T10 has the date of 2022-11-10. The range of dates is ten days  
Now we take a lambda function.   
This is a modified sigmoid and an extreme function showing date decay relevance.

x will be between 0 and 1. 0 for the oldest date and 1 for the most recent date. Between this range, it applies the lambda. In this case, lambda looks like that.

Chart, line chart

Description automatically generated

Plotted in desmos.com

We get the following weights for the transactions:

The function with parameters in the modified FP-growth gives the following result:

A screenshot of a computer

Description automatically generated with medium confidence

We see here that the equivalence on the transaction level is not destroyed. All items of the same transaction have the same weight. However, the frequent items and rules are now not an absolute frequency but an interpreted date decay frequency.

It seems very powerful; however, when not calibrated, especially on one database at a time, it will lead to misleading results. Therefore it should be used carefully.

Later transactions are still frequent, while items like I0, which was the most frequent in normal mode, are now ranked only as the third most frequent item and cannot build a frequent itemset with another item.

It is essential to mention that the date decay function dilutes the row count. We do not count every transaction anymore with 1, but the modified count after the date decay function.

## 4.4 Modified FP-growth with profit

Why does the author include profit as a metric inside the frequent items/ rules?  
Firstly, we can replace the decision with minimum profit instead of minimum support. There is somehow an equilibrium between more frequent, the same time cheaper, mostly higher margined items and expensive, less frequent, and lower margined items. We do not get the most relevant business rules if we consider frequency as a primary driver.

First use-case why profit as a metric will improve association rules:

1'000 items A with a profit of 10 equals 100 items B with a profit of 100. However, in traditional rules, one would never have a frequent association rule between both because the minimum support is probably chosen high enough that item B drops out. However, in sight of profit, they are equal, and in the author's opinion, they should both be selected. Instead, let us think about a very frequent item C, with 100,000 occurrences and 0.1 profit per item. Should it be selected? Most probably, it is irrelevant. In traditional association rules, it will spam everywhere and have "not relevant" associations.

Secondly, we solve a fundamental problem. Usually, TAR cannot handle transactions like:  
What would be the TAR outcome?

In TAR, the outcome would be very falsified. We could build subgroups for frequencies of items and create new items for these combinations, but that would dilute the item count.

If we consider profit as a second metric, we count it once as in TAR, but we add the profit to the item's overall profit; at the end, we calculate a new average profitability and create an optimal representation of the item's weight, not in the frequency of count, but in profit; it is like a shift from the strict count to the dynamic profit. It does not destroy association rules but represents their weight in a higher average profit.

For example, if one item always is bought five times, then at the end, the item's profit is five times higher for the single item, representing that it is, on average, bought five times in association rules

Surely in edge cases applying an average could be problematic if it is not representative; however, it is still better than simply not considering and counting only once.

Let us see how it works with the simple dataset.  
The author enriched the items with profit data, I0 with 10, I1 with 20, ..., I9 with 100.

Text

Description automatically generated

Graphical user interface, text

Description automatically generated

The author modified the simple dataset with three times I8 in Transaction T9. Let us see how we handle this.

We use the following parameters:  
Text

Description automatically generated

That will give us back the following result:

A screen shot of a computer

Description automatically generated with low confidence

A black screen with white text

Description automatically generated with low confidence  
  
The minSupRatio is not anymore the minimum frequency but is calculated in this way  
It can be interpreted as the most negligible percentage of the profit of a single item or association we still see as relevant. The descending order is defined according to the profit associated. The count and FP-tree construction are strictly according to traditional rules. We do not break the algorithm but enrich it.

Let us recalculate the profit associated with I8.  
The count of I8 is 5. That is the same as in TAR.  
In T9, we have the item 3 times.  
In total, the item count is, therefore, 8.  
The item's profit is 90.   
Let us recalculate the average profit of the I8:

For I8, as a single item without any other items combined, we have count five and an associated profit of 720 (=6 \*120). That represents the "real weight" of the item by profit; in frequency alone, it is not reflected.

Secondly, let us have a look at frequent itemset .  
The support for the itemset is 2.

That would fall out in the TAR because of low frequency. However, when we consider the combined associated profit, this combination will still be relevant for business.

Profit associated\_prev represents the total before the newly added item. If the new association has a higher profit than the previous one, it shows synergies. If it is lower, it is potentially nothing to focus on to " maximize" the overall profit.

## 4.5 Modified FP-growth with combined date decay and profit

When we combine the date decay and the profit function, we have the full range of parameters.

Text

Description automatically generated

Graphical user interface, text

Description automatically generated with medium confidence

Graphical user interface, text

Description automatically generated

According to classical TAR, the support (calculated) will be the same as in chapter 4.2. However, the fundamental selection of the "min support" items is chosen according to 4.3. Therefore, we do not find the same elements anymore. At the same time, the profit view gives us the great advantage of equalizing multiple items over the profit measure.  
The essential part and maybe a disadvantage of the custom date decay and profit lambda functions is the correct representative function for each dataset. Profit at easiest and most logically will be linear; however, there could be other means of measurement, which do not strictly measure profitability as such, but like a modified profit as a means of marginal profitability; for example, as a decreasing curve.

## 4.6 Performance, Strengths, and weaknesses of the author's modified FP-growth algorithm

The author checked the performance of the cosmetics dataset with a limitation of transactions no longer than five items as follows, and 77'264 (155'617 original) transactions, 40'777 unique items, support of 0.0001 (resulting in minimum support of 16) with 2'567 found itemsets/rules. Please notice that loading the prepared dataset in memory (dataframe) is omitted, which takes a big part of the work with smaller datasets.

Modified FP-growth (normal parameters): 14.1 seconds  
Modified FP-growth (profit parameters): around the same, 14.3 seconds  
FP-growth library from mlxtend: 5.0 seconds  
A-priori library from mlxtend: A-priori failed with minimum support of 0.0001 due to insufficient memory in the pandas dataframe. After decreasing minimum support to 0.001, it succeeded but needed 142.2 seconds, while the same, minor minimum support in FP-growth needed only 3.1 seconds.

We see that in a real-world dataset, A-priori becomes unusable and can fail due to memory limitations. That is especially true with the very diverse and rich cosmetics dataset, with many different, low-frequent items.

The author has to admit that the implementation of the FP-growth library from mlxtend is highly efficient, and the own implementation lacks some code optimizations. In academia, to explain fundamental concepts, the A-priori algorithm is much easier to grasp and modify. However, without digging deeper into performance, it can be concluded that every real-world algorithm for business should be mandatorily implemented as a variant of the FP-growth algorithm.

The strengths of the modified FP-growth algorithm see the author as follow:

* The single-path approach increases efficiency and gives more clarity about the association rules, meaning obsolete rules are not repeated, but the "most relevant" path is shown.
* The date-decay function is a feature that can be used in specific datasets; however, it is not recommended to use it in general. It is a highly specialized modification, which can make sense. The usefulness is, therefore, somewhat limited.
* Profit-function solves the problem of the correct weight of multiple items in one transaction; which otherwise gets lost
* By the profit function, in the author's opinion, we have better filtering criteria to find relevant datasets and associations. Moreover, the associated profit of itemsets is an interesting metric for profit-driven businesses.

The weaknesses of the modified FP-growth algorithm see the author as follow:

* When the author speaks of profit, we generally assume 10% of revenue. That simple approach may be enough; instead of profit, revenue can be chosen. However, to find the actual profit, the author would recommend determining the average profit per item category; in such a way, it will be better and more powerful. However, without insider data to an external dataset, it is nearly impossible to assume correct profit margins; or it is very time-consuming, which would not be worth it.
* In the date-decay function, we manipulate the support, and the meaning of support becomes different. Support of 1 does not mean one transaction but could be three transactions together, weighted by the date-decay function. Therefore that modification should be applied only for exceptional cases.
* We lose association rules by finding only one potentially relevant path. However, there are pros and cons; situational; however, it could have disadvantages
* The data preparation to get profit and date-decay functionality need more data pre-processing in the pandas dataframe; it has more parameters, meaning it is more difficult to use.

5 Experiments/ Analysis of big datasets

Let us experiment with the same datasets and compare the new profit-based AR algorithm with the traditional one.

## 5.1 E-Commerce purchase history from an electronic store

### 5.1.1 Dataset

Let us observe the dataset "E-Commerce purchase history from electronic store" [2] from Kaggle. It contains more than 2.6 million rows of item transactions. Each item is linked together either as a single transaction or with other items together by the "order\_id." The dataset has 25'113 unique items.

An essential part and maybe a disadvantage of the custom date decay and profit lambda functions is the correct representative function for each dataset. Profit at easiest and most logically will be linear; however, there could be other means of measurement, which do not strictly measure profitability as such, but like a modified profit as means of marginal profitability; for example, as a decreasing curve.

Graphical user interface, application, Teams

Description automatically generated

In the dataset, more than 60% are single-item transactions, therefore not relevant in finding association rules. However, their support still counts inside the minimum support. Around 20% of the dataset are 2-item transactions, and around 10% are 3-item transactions; the residual 10% is split in small numbers until 60 items of transactions are outliers.

### 5.1.2 Analysis

The dataset is not the most ideal for finding association rules. One reason could be that it is an online store, and most items are "higher priced." A client does not buy goods like in a grocery store. However, it is an excellent start to analyze. Later, let us compare a grocery store database and see the differences.

Here are the top 12 single items/itemsets found by classical association rules:

Graphical user interface, text

Description automatically generated

If we observe them, their data is either incomplete or does not have a price. However, from the prices, we generally can say that these are highly frequent, but

generally low prices articles. Are we interested in them? The author would assume they do not drive the business and are like "spam" in recognizing the relevant rules.

Conversely, when we let the association rules activate with profit measure, we find the following 12 top items/itemsets.

Graphical user interface, text, calendar

Description automatically generated

The only familiar item in line with traditional association rules is the item with rule 6. It is a Samsung smartphone with a relatively low price but high frequency, which is still relevant in total profit. Generally, when we observe the list, it looks much more relevant than the previous one. According to our expectation, when we would not know anything about consumer electronic goods, smartphones of different brands will be the top sellers/ top articles for stores. That already has some proof that the method with profit considered is superior.

Let us observe the rules, which have at least an item set of articles. Here are the unique and relevant rules to observe.

Graphical user interface, text

Description automatically generated with medium confidence

Itemsets 1 & 2 from Grohe are very popular. Grohe belongs to plumbing, an necessary, but rather a B2B good instead of B2C. The installation is rarely done by end customers but by plumbers. However, the "systematic" purchase of the same plumbings as itemsets lets us assume that the purchases are very planned and will have a strong pattern. Interestingly the three itemset of Grohe has no more substantial profit than the itemset of 2 Grohe articles due to the much smaller standard frequency and the smaller price of the third article. Itemset 3 is very interesting; goodride (car products) and dogland (dog pillows) could make sense that it is a complimentary set to keep a dog in the car. It is bought frequently, at a moderate price, making much profit. Itemsets 4, 5, and 6 are Smartphones with a high price (primary product) and a complimentary product like a smartphone cover. It makes absolute sense to buy these items together. However, it makes sense that Smartphones are mostly bought alone (single items) and that two smartphones (basic) products are not bought together as a single transaction.

## 5.2 E-Commerce purchase history from a jewelry store

### 5.2.1 Dataset

The Kaggle provider (3) of this dataset states, "This file contains purchase data from December 2018 to December 2021 (3 years) from a medium-sized jewelry online store."

The dataset has 95'911 rows. It has 74'760 transactions. Thereof are 9'613 unique items. 2'298 transactions contain multiple items. The TAR is not able to solve this discrepancy. However, with our improved algorithm, we can balance it out by the profit measure. The distribution of multiple items per transaction is shown below:

Chart, histogram

Description automatically generated

The length of transactions is distributed like that:

Graphical user interface, application, Teams

Description automatically generated

In the dataset, more than 80% are single-item transactions, therefore not relevant in finding association rules. However, their support still counts inside the minimum support. Around 15% of the dataset are 2-item transactions, and around 3% are 3-item transactions; the residual 2% is split in small numbers until 27 items of transactions are outliers. Reasonably, this pattern is expected because jewelry is a luxury good and is not expected to be bought as would be goods in a grocery store.

### 5.2.2 Analysis

The second dataset is even worse than the first for finding association rules. The reason this time is quite clear. Jewelry is a luxury not that often bought in a single transaction.

Here are the top 12 single items/itemsets found by classical association rules:

Graphical user interface, text

Description automatically generated

All of them are single transactions. We can observe that the most common items are relatively low priced compared to an average item price of 405.49 and in the lower half compared to the median of 261.60.

Next, let us observe 15 rows of only association rules with at least two items.

Graphical user interface, text

Description automatically generated

Interestingly the price of the itemset pair has some positive correlation, and the price range of these item pairs is relatively homogenous. Unfortunately, we do not have more specific data about the product, but it seems like either the jewelry itemsets are in a collection of a brand or maybe something very classical wedding ring; a woman's ring is higher priced with a gem, while a man's ring lower priced without a gem. What we are missing here in the traditional Association rules are the high-priced combinations, which are expected to be much lower in joint support, but will be relevant because of the overall price. Moreover, we do not consider items bought in a quantity of more than 1. All found items are rings or earrings. It seems to be the most popular overall item.

Let us now observe profit-based AR.

Graphical user interface, text

Description automatically generated

The top 15 profitable itemsets are single-item transactions, the same as in TAR. However, the price is either on the upper side of the median or the opposite of TAR. The support on the other side is very homogenous, ranging from 5 to 301. The item with only five support is still relevant because of its high price of 26'425. The item with 301 has the lowest price of 260. We see similarly that rings and earrings are popular, each in support and profitability. That is different from the electronic goods store in the first analysis, where we found utterly different kinds of items. However, one must admit that the jewelry category range is small. That could bring a reasonable explanation. What is unclear is the "nan" category. The author assumed these are jewelry adjustments like adjusted size, extra gems, and custom services. The intersection of items with the TAR is two items out of 16.

Let us move to at least 2 item transactions.

Graphical user interface, text

Description automatically generated with medium confidence

The result is surprising. The results of TAR correlate quite a bit with profit-based AR. The top 2 itemsets are identical. Out of 8 itemsets, three are identical to AR.

What pitches the eye are the itemsets with only seven support but are highly priced. These items are pretty relevant even if the support is low. Surely there should be a hard lower threshold of irrelevance. Below support of 5, maybe, even if the price is high, the relevance is diminished because of randomness. A rule would not be strict enough and be evident to predict this relation for future transactions.

In this jewelry dataset, the advantages of a profit AR are lower than in the consumer electronic stores. The author concludes this because of the much higher homogeneity of jewelry products.

## 5.3 E-Commerce purchase history from a cosmetics store

### 5.3.1 Dataset

The Kaggle provider [4] of this dataset states: "This file contains behavior data for five months (Oct 2019 – Feb 2020) from a medium cosmetics online store. Each row in the file represents an event. All events are related to products and users. Each event is like many-to-many relations between products and users." Originally the dataset contained the events view, cart, remove\_from\_cart, and purchase. For the author's analysis, only purchases are considered.

The dataset has 1'287'007 rows. It has 155'617 transactions. Thereof are 40'777 unique items. 10'123 transactions contain multiple items. The TAR is not able to solve this discrepancy. However, with our improved algorithm, we can balance it out by the profit measure. The distribution of multiple items per transaction is shown below:

Chart

Description automatically generated

Chart

Description automatically generated with medium confidence

The cosmetics dataset is quite different from the previous ones. It has a relatively low number of single-item transactions. Cosmetics are most frequently bought together as a transaction set of 3 items with a frequency of around 11%. The curve is relatively flat, meaning there are a lot of combined items per transaction. That is ideal for finding association rules. One can expect that distribution because cosmetics are cheaper than electronics and jewelry. Moreover, in cosmetics, we will have many complementary products bought together.

### 5.2.2 Analysis

Here are the top 10 single itemsets found by classical association rules:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| idx | itemset | support | product\_id | category\_id | brand | price |
| 0 | 5809910 | 7458 | 5809910 | 1602943681873052386 | grattol | 5.24 |
| 1 | 5854897 | 4593 | 5854897 | 1487580009445982239 | irisk | 0.32 |
| 2 | 5700037 | 3623 | 5700037 | 1487580009286598681 | runail | 0.4 |
| 3 | 5802432 | 3486 | 5802432 | 1487580009286598681 | - | 0.32 |
| 4 | 5751422 | 3457 | 5751422 | 1487580005268456287 | uno | 10.95 |
| 5 | 5809912 | 3272 | 5809912 | 1602943681873052386 | grattol | 5.24 |
| 6 | 5815662 | 3206 | 5815662 | 1487580006317032337 | - | 0.92 |
| 7 | 5304 | 3102 | 5304 | 1487580009471148064 | runail | 0.32 |
| 8 | 5751383 | 2912 | 5751383 | 1487580005092295511 | uno | 10.32 |
| 9 | 5849033 | 2753 | 5849033 | 1487580005092295511 | uno | 10.32 |

None of the associated itemsets (2-itemsets) got into the top 10. The image is very homogenous, ranging from prices from 0.32 to 10.32, support from 2'753 (1.77%) to 7'458 (4.79%). For such an extensive dataset, these are relatively high numbers. The most popular item category from brand names seems to be nail polish and nail care products. From a price perspective, these items represent the median, which is 4.44 for the whole dataset, while the mean is 7.05.

Let us observe the top 11 itemsets in profit-based AR:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| idx | itemset | support | product\_id | category\_id | brand | price |
| 0 | 5560754 | 187 | 5560754 | 1487580006300255120 | strong | 194.44 |
| 1 | 89343 | 47 | 89343 | 2193074740619379535 | nan | 299.81 |
| 2 | 5877454 | 315 | 5877454 | 1487580006300255120 | jessnail | 44.29 |
| 3 | 5850281 | 90 | 5850281 | 1487580006300255120 | marathon | 137.78 |
| 4 | 5692527 | 256 | 5692527 | 1487580014009385185 | nan | 46.02 |
| 5 | 5856186 | 150 | 5856186 | 1487580006300255120 | runail | 75.4 |
| 6 | 5846437 | 248 | 5846437 | 1487580013950664926 | browxenna | 45.24 |
| 7 | 5560756 | 50 | 5560756 | 1487580006300255120 | strong | 207.94 |
| 8 | 5855507 | 125 | 5855507 | 1487580006350586771 | max | 79.21 |
| 9 | 5804820 | 138 | 5804820 | 1487580013950664926 | nan | 63.33 |
| 10 | 5809910 | 1642 | 5809910 | 1602943681873052386 | grattol | 5.24 |
| 11 | 5861760 | 116 | 5861760 | 1487580006350586771 | emil | 71.43 |

The price ranges from 5.24 to 299.81. Compared to TAR, it does not look homogenous. Instead, we have an equilibrium on profitability. The cheap product, with a price of 5.24, makes a similar profit as the expensive product, with a price of 299.81.

Here are the top 5 itemsets with at least two items in TAR:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| idx | itemset | support | product\_id | category\_id | brand | price |
| 0 | 5809910 | 231 | 5809910 | 1602943681873052386 | grattol | 5.24 |
| 0 | 5809912 | 231 | 5809912 | 1602943681873052386 | grattol | 5.24 |
| 1 | 5751383 | 175 | 5751383 | 1487580005092295511 | uno | 10.32 |
| 1 | 5751422 | 175 | 5751422 | 1487580005268456287 | uno | 10.95 |
| 2 | 5809910 | 161 | 5809910 | 1602943681873052386 | grattol | 5.24 |
| 2 | 5809911 | 161 | 5809911 | 1602943681873052386 | grattol | 5.24 |
| 3 | 5814516 | 148 | 5814516 | 1487580013539623112 | nan | 13.49 |
| 3 | 5814517 | 148 | 5814517 | 1487580013539623112 | nan | 12.54 |
| 4 | 5697463 | 129 | 5697463 | 1783999067156644376 | estel | 12.14 |
| 4 | 5677043 | 129 | 5677043 | 1487580008246412266 | estel | 13.33 |

The itemsets are very interesting. They are different products but have the same brand and prices. It can be assumed that these items are like a variant of each other, like a change of nail polish or lipstick color. The prices are standard compared to the median and average; for shelf management, these products should be presented directly to each other or, in e-commerce, should be recommended in an online store.

Here are the top 5 itemsets with at least two items in profit-based AR.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| idx | itemset | support | | product\_id | category\_id | brand | price |
| 0 | 5814516 | | 148 | 5814516 | 1487580013539623112 | nan | 13.49 |
| 0 | 5814517 | | 148 | 5814517 | 1487580013539623112 | nan | 12.54 |
| 1 | 5751383 | | 175 | 5751383 | 1487580005092295511 | uno | 10.32 |
| 1 | 5751422 | | 175 | 5751422 | 1487580005268456287 | uno | 10.95 |
| 2 | 3762 | | 88 | 3762 | 1487580005411062629 | cnd | 19.37 |
| 2 | 4185 | | 88 | 4185 | 1487580005411062629 | cnd | 19.37 |
| 3 | 5697463 | | 129 | 5697463 | 1783999067156644376 | estel | 12.14 |
| 3 | 5677043 | | 129 | 5677043 | 1487580008246412266 | estel | 13.33 |
| 4 | 5809910 | | 231 | 5809910 | 1602943681873052386 | grattol | 5.24 |
| 4 | 5809912 | | 231 | 5809912 | 1602943681873052386 | grattol | 5.24 |

Interestingly, profit-based AR delivers nearly precisely the same itemsets. Only the itemset with 88 support and a higher price is different. One can assume that high-priced cosmetics are not bought in frequent combinations with other products, and regular products have a quite homogenous price, resulting in nearly the same outcome.

6 Conclusion

In chapter 1, we saw much research done on association rules. However, the critical problem of successful implementation for businesses remains. The author addressed four fundamental problems to which he wants to suggest solutions.

1. Finding relevant rules based on a secondary profit metric instead of frequency
2. Consider Transactions with multiple same items
3. Date-Decay function as a feature for exceptional cases
4. Increase clarity of AR by reducing itemset's AR to the most frequent path

In chapter 2, we discussed related work in the field of AR. The starting point of AR was already in 1993 when Agrawal et al. [1]. wrote a paper about association rules. Since then, the research has flourished, and all research can be categorized into four categories [9]:

1. Reducing the number of passes over the database
2. Sampling the database
3. Adding extra constraints on the structure of patterns
4. Increase efficiency through parallelization

The author's proposed solutions belong to category 3, in the subcategory of the processing inside the algorithm, opposite pre- or post-processing.

In chapter 3, the author introduced an exemplary sample dataset and showed the A-priori and the baseline FP-growth algorithm step by step. We have seen that A-priori is quite a simple algorithm, while FP-growth is more complex to implement.

In chapter 4, the author introduced the modified FP-growth algorithm and explained the differences from the baseline algorithm. After that, examples and experiments based on the sample dataset are shown. It is shown how the four problems we stated in the introduction the author intends to solve with his modified algorithm. At last, the performance on an extensive dataset is compared among A-priori, FP-growth, and the author's modified FP-growth. Both FP-growth and modified FP-growth perform similarly; however, FP-growth from mlxtend is the clear winner. The author has not made all the necessary optimization steps. However, A-priori was ten times slower than the author's modified FP-growth. The time complexity of A-priori is O(2n), compared to FP-growth's O(n2).

In chapter 5, 3 datasets have been analyzed to show the advantages of the modified FP-growth algorithm. These three datasets were quite different:

- E-Commerce purchase history from the electronic store: Quite homogenous items, high variance in prices. Items are most often purchased as single-item transactions.

- E-Commerce purchase history from jewelry store: Quite similar items, but different in price. Items are most often purchased as single-item transactions.

E-Commerce purchase history from cosmetics store: Quite similar items, but different in price. Items are most often purchased as multi-item transactions.

The results are interesting and show that for the cosmetics store, we found very similar top itemsets with TAR and modified FP-growth, while in the electronic good and jewelry, we found some overlapping but included the higher priced, average frequent items and skipped the highly frequent but very low priced items.

In the author's opinion, the experiments successfully showed an improvement in finding relevant and strong itemsets and association rules.

Future research in implementing especially a profit-metric as filter criteria and solving the multi-item transaction can further focus on materializing and measuring the benefit of such an approach and further modify the function to standardize a proven approach for businesses in the real world. Moreover, the implementation of the modified FP-growth can be further optimized.

7 References

1. Agrawal, R.; Imieliński, T.; Swami, A. (1993). Mining association rules between sets of items in large databases. Proceedings of the 1993 ACM SIGMOD international conference on Management of data - SIGMOD '93. p. 207
2. E-Commerce purchase history from electronic store (Retrieved: 02.10.2022 from https://www.kaggle.com/datasets/mkechinov/ecommerce-purchase-history-from-electronics-store)
3. E-Commerce purchase history from jewelry store (Retrieved: 11.12.2022 from https://www.kaggle.com/datasets/mkechinov/ecommerce-purchase-history-from-jewelry-store)
4. E-Commerce purchase history from cosmetics shop (Retrieved: 12.12.2022 from https://www.kaggle.com/datasets/mkechinov/ecommerce-events-history-in-cosmetics-shop)
5. Kotsiantis, S.; Kanellopoulos, D. (2006). Association Rules Mining: A Recent Overview. GESTS International Transactions on Computer Science and Engineering. Vol.32 (1), 2006, pp. 71-82
6. Han, J. and Pei, J. (2000). Mining frequent patterns by pattern-growth: methodology and implications. ACM SIGKDD Explorations Newsletter 2, 2, 14-20.
7. Toivonen, H. (1996. Sampling large databases for association rule. The VLDB Journal, pp. 134-145.
8. Das, A., Ng, W.-K., and Woon, Y.-K. 2001. Rapid association rule mining. Proceedings of the tenth international conference on information and knowledge management. ACM Press, 474-481
9. Slimani, T., Lazzez, A. 2014. Efficient Analysis of Pattern and Association Rule Mining Approaches. International Journal of Information Technology and Computer Science (IJITCS), vol.6, no.3, pp.70-81, 2014
10. Patel, H, K., Yadav, K, P. An Innovative Approach for Association Rule Mining in Grocery Dataset Based On Non-Negative Matrix Factorization And Autoencoder. Journal of Algebraic Statistics, Volume 13, No. 3, 2022, p. 2898 – 2905
11. Al Shehabi, S., Baba, A. 2021. MARC: Mining Association Rules from datasets by using Clustering models. Published: 2021-07-14, Retrieved 2022-09-22 from: https://arxiv.org/abs/2107.08814
12. Bao, F., Mao, L., Zhu, Y., Xiao, C., Xu, C. 2021. An Improved Evaluation Methodology for Mining Association Rules. Axioms, Published: 2021-12-30, Retrieved from https://www.mdpi.com/2075-1680/11/1/17
13. Tahyudin, I., Haviluddin, I,. Nanbo, H. 2019. Time Complexity Of A Priori And Evolutionary Algorithm For Numerical Association Rule Mining Optimization. International Journal of Scientific & Technology Research Volume 8, Issue 11, November 2019
14. Retrieved 2022-12-22 from: https://stackoverflow.com/questions/9869884/what-is-the-time-and-space-complexity-of-FP-growth-algorithm
15. Chou, T., 2020, FP Growth - Frequent Pattern Generation in Data Mining with Python Implementation, Retrieved 2022-10-23 from: https://towardsdatascience.com/FP-growth-frequent-pattern-generation-in-data-mining-with-python-implementation-244e561ab1c3

8 Applications

Datasets, Algorithms, Analyses, and Master Thesis are saved in the author's repository: https://github.com/dawei7/Detecting-patterns-in-purchase-history-using-association-rule-learning-methods